

PRAKTIKUM PEMROGRAMAN JARINGAN
JOBSHEET 3
“Node.Js Module System”



DOSEN PENGAMPU:
Randi Proska Sandra, M.Sc.

OLEH:
Muhammad Alfarobi
23343011

PROGRAM STUDI INFORMATIKA
DEPARTEMEN TEKNIK ELEKTRONIKA
FAKULTAS TEKNIK
UNIVERSITAS NEGERI PADANG
2025

LATIHAN

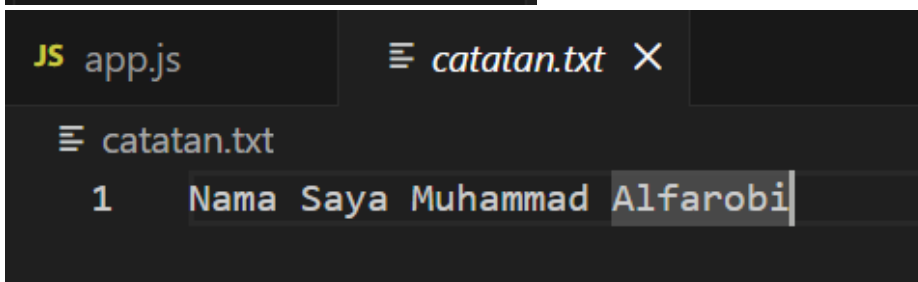
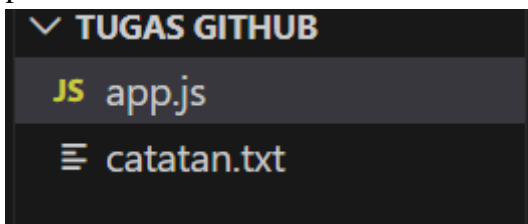
A. NODEJS MODULE SYSTEM

a. Import Core Module NodeJS

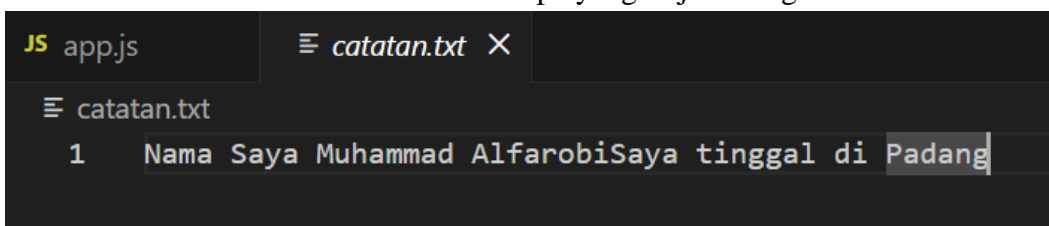
1. Buatlah sebuah folder baru dengan nama buku-catatan dan dalam folder tersebut buatlah file javascript baru dengan nama app.js dan ketikkanlah kode berikut ini

```
JS app.js > ...  
1  const fs = require('fs')  
2  
3  fs.writeFileSync('catatan.txt', 'Nama Saya Muhammad Alfarobi'  
4  //fs.appendFileSync('catatan.txt', 'Saya tinggal di Padang')  
5  )
```

2. Jalankan source code tersebut melalui terminal pada visual studio code dan perhatikan sebuah file baru bernama catatan.txt telah dibuat.



3. Jadikan baris fs.writeFileSync sebagai komentar dan kemudian uncomment baris fs.appendFileSync.
4. Jalankan Kembali source code dan lihat apa yang terjadi dengan catatan.txt



b. Import File Pada NodeJS

1. Buatlah file baru pada folder buku-catatan dengan nama catatan.js dan ketikkan kode berikut ini

```
const ambilCatatan = function () {  
  return 'Ini Catatan Muhammad Alfarobi...'  
}  
module.exports = ambilCatatan
```

2. Lalu ketikkan kode berikut pada file app.js

```
const catatan = require('./catatan.js')
const pesan = catatan()
console.log(pesan)
```

3. Jalankan program melalui terminal dan perhatikan apa yang ditampilkan

c. Import npm Pada NodeJS

1. Silakan buka website <https://www.npmjs.com/> untuk menemukan package yang akan digunakan
2. Buka terminal pada visual studio code dan pastikan anda berada pada direktori buku-catatan. Lalu ketikkanlah npm init dan tekan enter hingga muncul seperti pada gambar dibawah ini

```
{
  "name": "buku-catatan",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit"
  },
  "author": "robi ",
  "license": "ISC",
  "type": "commonjs"
}
```

3. Setelah selesai akan muncul file baru pada direktori buku-catatan anda dengan nama package.json
4. Carilah package validator pada website npm. Validator adalah library untuk validasi string dan anda dapat menemukan informasi lengkapnya pada website npm, termasuk bagaimana cara menginstall dan menggunakannya
5. Lakukan instalasi package tersebut melalui terminal dengan perintah npm i validator@13.11.0. Penambahan @13.11.0 pada akhir perintah instalasi bertujuan untuk menginstall package sesuai dengan versi yang diinginkan
6. Perhatikan bahwa sebuah folder baru node-modules telah muncul pada direktori anda. Selain itu, juga ada file baru dengan nama package-lock.json yang berisi informasi terkait package yang diinstall
7. Ketikkanlah kode berikut pada file app.js dan jalankan program melalui terminal

```
const validator = require('validator')
const catatan = require('./catatan.js')
const pesan = catatan()
console.log(pesan)
console.log(validator.isURL('https://robi.com'))
```

```
{
  "name": "buku-catatan",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit"
  },
  "author": "robi ",
  "license": "ISC",
  "type": "commonjs"
}
```

d. Latihan 1

1. Lakukan instalasi package chalk versi 4.1.2 pada direktori aplikasi buku-catatan
2. Buatlah perintah untuk mencetak teks berikut menggunakan chalk 'print warna biru sukses'. Perhatikan dokumentasi chalk pada website npm <https://www.npmjs.com/package/chalk/v/4.1.2> untuk mengetahui bagaimana penggunaannya.
3. Penting! Pastikan anda menambahkan variable const chalk pada awal program
4. Cobalah untuk mengganti warna teks biru dengan warna dan variasi lainnya

```
const chalk = require('chalk')
const validator = require('validator')
const catatan = require('./catatan.js')
const pesan = catatan()
console.log(pesan)
console.log(validator.isURL('https://robi.com'))
console.log(chalk.blue(pesan))
console.log(chalk.red.italic.bold(pesan))
)
```

```

catatan > package.json > ...
{
  "name": "buku-catatan",
  "version": "1.0.0",
  "description": "",
  "license": "ISC",
  "author": "robi",
  "main": "app.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "dependencies": {
    "chalk": "^4.1.2",
    "validator": "^13.15.15"
  }
}

```

e. Latihan 2

1. Lakukan instalasi package nodemon versi terbaru dengan perintah `npm install nodemon@[versi package terbaru] -g`
2. Cek apakah nodemon berhasil terinstall dengan perintah `nodemon -v` pada terminal
3. Jika tidak muncul informasi versi nodemon, maka jalankan Windows PowerShell dan ketikkan perintah `Set-ExecutionPolicy Unrestricted`, lalu coba jalankan lagi perintah `nodemon -v`
4. Apabila sudah terinstall dengan rapi, jalankan aplikasi anda menggunakan perintah `nodemon`, bukan lagi `node`. Contoh: `nodemon app.js`
5. Ubahlah salah satu coding program anda dan perhatikan bahwa program anda yang berjalan akan otomatis berubah tampilannya secara realtime

```

const chalk = require('chalk')
const validator = require('validator')
const catatan = require('./catatan.js')
const pesan = catatan()
console.log(pesan)
console.log(validator.isURL('https://robi.com'))
console.log(chalk.blue(pesan))
console.log(chalk.red.italic.bold(pesan))
console.log(chalk.green.underline.bold(pesan))

```

```

catatan > package.json > ...
{
  "name": "buku-catatan",
  "version": "1.0.0",
  "description": "",
  "license": "ISC",
  "author": "robi",
  "main": "app.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "dependencies": {
    "chalk": "^4.1.2",
    "validator": "^13.15.15"
  },
  "devDependencies": {
    "nodemon": "^3.1.10"
  }
}

```

B. COMMAND LINE ARGUMENTS

a. Mendapatkan Input Dari Pengguna

1. Ketikkanlah kode berikut pada file app.js anda
2. Jalankan pada terminal dengan perintah node app.js [tambahkan argument anda dibelakang]. Misal: node app.js Randi Proska Sandra
3. Lalu, cobalah mengganti nilai array[] pada baris pertama, misal dengan angka 5. Anda juga bisa memasukan nilai array[] pada baris kedua dengan memberikan perintah seperti berikut
4. Perhatikan dan pahami apa yang ditampilkan
5. Uncomment source code yang ada pada komentar, lalu jalankan melalui terminal dengan perintah node app.js tambah atau node app.js hapus
6. Perhatikan dan pahami apa yang ditampilkan

```
console.log(process.argv)

if(command === 'tambah') {
  console.log('Tambah data catatan')
} else if(command === 'hapus') {
  console.log('Hapus data catatan')
}
```

```
"main": "app.js",
  > Debug
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "dependencies": {
    "chalk": "^4.1.2",
    "validator": "^13.15.15"
  },
  "devDependencies": {
    "nodemon": "^3.1.10"
  }
}
```

b. Argument Parsing

1. Lakukan instalasi package yargs pada direktori aplikasi buku-catatan. Yargs adalah satu package pada javascript yang berguna untuk argument parsing.
2. Masukkanlah kode berikut pada file app.js
3. Jalankan program diatas melalui terminal pada visual studio code dengan perintah node app.js tambah atau node app.js hapus. Perhatikan dan pahami apa yang ditampilkan

```
const yargs = require('yargs')
const catatan = require('./catatan.js')
// yargs.version('10.1.0')

yargs.command({
  command: 'tambah',
  describe: 'Menambahkan catatan baru',
  handler: function() {
    console.log('Menambahkan data catatan')
  }
})

yargs.command({
  command: 'hapus',
  describe: 'Menghapus catatan',
  handler: function() {
    console.log('Menghapus data catatan')
  }
})

console.log(yargs.argv)
```

```
"main": "app.js",
  > Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "dependencies": {
    "chalk": "^4.1.2",
    "validator": "^13.15.15",
    "yargs": "^16.2.0"
  },
  "devDependencies": {
    "nodemon": "^3.1.10"
  }
}
```

4. Jika ini sebuah program yang lengkap dan memiliki banyak jumlah catatan didalam aplikasinya, maka diperlukan perintah untuk membaca catatan dan perintah untuk menampilkan semua catatan. Lanjutkan program diatas dengan menambahkan perintah (command) untuk menampilkan list (daftar) semua catatan serta perintah untuk membaca (read) sebuah catatan
5. Setelah anda menambahkan command pada nomor 4, ubahlah coding pada command tambah dengan source code dibawah ini
6. Lalu ubahlah baris terakhir pada program dengan code
7. Jalankan file app.js melalui terminal dengan perintah node app tambah -- judul="Catatan Randi", --isi="Ini adalah catatan pertama Randi"
8. Perhatikan dan pahami apa yang ditampilkan

```

const yargs = require('yargs')
const catatan = require('./catatan.js')
// yargs.version('10.1.0')

yargs.command({
  command: 'tambah',
  describe: 'Menambahkan catatan baru',
  builder: {
    judul: {
      describe: 'Judul catatan',
      demandOption: true,
      type: 'string'
    },
    isi: {
      describe: 'Isi catatan',
      demandOption: true,
      type: 'string'
    }
  },
  handler: function(argv) {
    console.log('Judul : ' + argv.judul)
    console.log('Isi : ' + argv.isi)
  }
})

yargs.command({
  command: 'hapus',
  describe: 'Menghapus catatan',
  handler: function() {
    console.log('Menghapus data catatan')
  }
})

yargs.command({
  command: 'lihat',
  describe: 'Melihat catatan',
  handler: function() {
    console.log(catatan())
  }
})

yargs.parse()

```

```

"main": "app.js",
  > Debug
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "dependencies": {
    "chalk": "^4.1.2",
    "validator": "^13.15.15",
    "yargs": "^16.2.0"
  },
  "devDependencies": {
    "nodemon": "^3.1.10"
  }
}

```

c. Menyimpan Data dengan JSON

1. Buatlah folder baru dengan nama testsite, lalu buatlah file baru dalam folder tersebut dengan nama 1-jsontest.js
2. Ketikkanlah source code berikut pada file tersebut
3. Perhatikan dan pahami apa yang terjadi.


```

catatan > testsite > JS 1-jsontest.js > ...
const fs = require('fs')

const buku = {
  judul: 'Belajar Node.js',
  penulis: 'Muhammmad Alfarobi',
}

const bookJSON = JSON.stringify(buku)
fs.writeFileSync('1-buku.json', bookJSON)
}

```

d. Membaca Data JSON

1. Jadikan coding diatas menjadi komentar, kecuali baris pertama, lalu masukanlah code berikut ini
2. Jalankan program app.js, lalu perhatikan dan pahami apa yang ditampilkan!

```

const dataBuffer = fs.readFileSync('1-buku.json')
const dataJSON = dataBuffer.toString()
const buku = JSON.parse(dataJSON)
console.log(buku)

```

3. Ubahlah console.log(data) menjadi console.log(data.judul), jalankan dan perhatikan perbedaan tampilannya dengan Langkah sebelumnya.

```

$ node 1-jsontest.js
{ judul: 'Belajar Node.js', penulis: 'Muhammad Alfarobi' } }

```

e. Latihan menyempurnakan aplikasi buku catatan

1. Menambahkan catatan baru
 - a. Perhatikan Langkah pada latihan bagian b nomor 5. Ubahlah kode pada bagian handler menjadi seperti berikut

```

handler: function(argv) {
  catatan.tambahCatatan(argv.judul, argv.isi)
}

```

- b. Ubahlah kode pada file catatan.js menjadi berikut ini

```

buku-catatan > JS catatan.js > ...
1  const fs = require('fs')
2  const importNote = function () {
3    return 'Ini Catatan FPG...'
4  }
5  const addNote = function (judul, isi) {
6    const catatan = loadNote()
7    const catatanGanda = catatan.filter(function (note) {
8      return note.title === judul
9    })
10
11    if (catatanGanda.length === 0) {
12      catatan.push({
13        judul: judul,
14        isi: isi
15      })
16      saveNote(catatan)
17      console.log('Catatan baru ditambahkan!')
18    } else {
19      console.log('Judul catatan telah dipakai')
20    }
21  }
22  const saveNote = function (catatan) {
23    const dataJSON = JSON.stringify(catatan)
24    fs.writeFileSync('catatan.json', dataJSON)
25  }
26  const loadNote = function () {
27    try {
28      const dataBuffer = fs.readFileSync('catatan.json')
29      const dataJSON = dataBuffer.toString()
30      return JSON.parse(dataJSON)
31    } catch (e) {
32      return []
33    }
34  }
35  module.exports = {
36    importNote: importNote,
37    addNote: addNote
38  }

```

- c. Jalankan program diatas melalui terminal dengan perintah node app.js tambah -judul="Catatan 1", --isi="Isi catatan 1" dan perhatikan bahwa file baru catatan.json telah dibuat. Bukalah file tersebut dan lihat isinya

```

const yargs = require('yargs')
const catatan = require('./catatan.js')
// yargs.version('10.1.0')

yargs.command({
  command: 'tambah',
  describe: 'Menambahkan catatan baru',
  builder: {
    judul: {
      describe: 'Judul catatan',
      demandOption: true,
      type: 'string'
    },
    isi: {
      describe: 'Isi catatan',
      demandOption: true,
      type: 'string'
    }
  },
  handler: function(argv) {
    catatan.addNote(argv.judul, argv.isi)
  }
})

yargs.command({
  command: 'hapus',
  describe: 'Menghapus catatan',
  handler: function() {
    console.log('Menghapus data catatan')
  }
})

yargs.command({
  command: 'lihat',

```

```
$ node app.js tambah --judul="Catatan 1" --isi="Isi catatan 1"  
Catatan baru ditambahkan!
```

2. Menghapus catatan

- Bukalah coding pada file app.js anda dan perhatikan bahwa perintah (command) hapus anda tidak memiliki builder sebagaimana command tambah
- Buatlah builder untuk command hapus anda seperti pada command tambah namun cukup judul saja dan tanpa bagian isi.
- Untuk bagian handler, gunakanlah code berikut
- Tambahkan fungsi baru pada file catatan.js dan letakan dibawah fungsi tambahCatatan. Masukkan kode berikut!
- Kode diatas menggunakan perintah dari chalk pada bagian console.log, sehingga anda perlu menambahkan variabel chalk pada awal program di file catatan.js
- Selain itu, anda juga perlu memperbarui module.exports dengan memanggil fungsi untuk menghapus catatan
- Jalankan program diatas melalui terminal dengan perintah node app.js hapus --judul="Catatan 1"

```
yargs.command({  
  command: 'hapus',  
  describe: 'Menghapus catatan',  
  builder: {  
    judul: {  
      describe: 'Judul catatan',  
      demandOption: true,  
      type: 'string'  
    }  
  },  
  handler: function(argv) {  
    catatan.deleteNote(argv.judul)  
  }  
})  
  
yargs.command({  
  command: 'lihat',  
  describe: 'Melihat catatan',  
  handler: function() {  
    console.log(catatan())  
  }  
})
```

```
const deleteNote = function (judul) {  
  const catatan = loadNote()  
  const catatanUntukDisimpan = catatan.filter(function (note) {  
    return note.judul !== judul  
  })  
  if(catatan.length > catatanUntukDisimpan.length) {  
    console.log(chalk.green.inverse('Catatan dihapus!'))  
    saveNote(catatanUntukDisimpan)  
  } else {  
    console.log(chalk.red.inverse('Catatan tidak ditemukan!'))  
  }  
}
```

```
$ node app.js hapus --judul="Catatan 1"  
Catatan tidak ditemukan!
```

```
$ node app.js hapus --judul="Catatan 2"  
Catatan dihapus!
```

3. Lanjutkan program diatas untuk fungsi menampilkan semua catatan (list) dan membaca catatan (read)

```
    })  
    yargs.command({  
      command: 'lihat',  
      describe: 'Melihat catatan',  
      handler: function() {  
        catatan.listNote()  
      }  
    })  
  })  
  yargs.parse()
```

```
const listNote = function () {  
  const catatan = loadNote()  
  if (catatan.length === 0) {  
    console.log(chalk.red.inverse('Tidak ada catatan yang tersimpan.'))  
  } else {  
    console.log(chalk.cyan.bold('Daftar Catatan:'))  
    catatan.forEach((note, index) => {  
      console.log(chalk.yellow(`${index + 1}. ${note.judul}`))  
      console.log(chalk.white(`    ${note.isi}\n`))  
    })  
  }  
}
```

```
• $ node app.js lihat  
  Daftar Catatan:  
  1. Catatan 1  
    Isi catatan 1
```